Terms used
**branching** and **instruction** and **register** and **valid** and **check** and **processing** and **address** and **clock** and **cycle** and **guard** **architecture**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search
Try this search in The ACM G

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown Re

1 Beating in-order stalls with "flea-flicker" two-pass pipelining
Ronald D. Barnes, Erik M. Nystrom, John W. Sias, Sanjay J. Patel, Nacho Navarro, Wen-mei W. Hwu
December 2003 **Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitectu**

Full text available: pdf(247.31 KB) Publisher Site Additional Information: full citation, abstract, index terms

Accommodating the uncertain latency of load instructionsis one of the most vexing problems in in-order microarchitecturedesign and compiler development. Compilers cangenerate schedules with a high degree of in levelparallelism but cannot effectively accommodate unanticipatedlatencies; incorporating traditional out-of-o executioninto the microarchitecture hides some of this latencybut redundantly performs work done by the com additional pipeline stages. Although effectiv ...

2 Formal verification in hardware design: a survey
Christoph Kern, Mark R. Greenstreet
April 1999 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 4 Issue 2

Full text available: pdf(411.53 KB) Additional Information: full citation, abstract, references, citings, index terms

In recent years, formal methods have emerged as an alternative approach to ensuring the quality and correct designs, overcoming some of the limitations of traditional validation techniques such as simulation and testing main aspects to the application of formal methods in a design process: the formal framework used to specify properties of a design and the verification techniques and tools used to reason about the relationship between

**Keywords:** case studies, formal methods, formal verification, hardware verification, language containment, m survey, theorem proving

3 Pipeline Architecture
C. V. Ramamoorthy, H. F. Li
January 1977 **ACM Computing Surveys (CSUR)**, Volume 9 Issue 1

Full text available: pdf(3.53 MB) Additional Information: full citation, references, citings, index terms

4 Compiler transformations for high-performance computing
David F. Bacon, Susan L. Graham, Oliver J. Sharp
December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4

Full text available: pdf(6.32 MB) Additional Information: full citation, abstract, references, citings, index terms, revi

In the last three decades a large number of compiler transformations for optimizing programs have been impl optimizations for uniprocessors reduce the number of instructions executed by the program using transformat

h c g e cf c

the analysis of scalar quantities and data-flow techniques. In contrast, optimizations for high-performance sup and parallel processors maximize parallelism and memory locality with transformations that rely on tracking t o ...

**Keywords**: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism, superscala vectorization

5 A survey of processors with explicit multithreading
Theo Ungerer, Borut Robič, Jurij Šilc
March 2003 **ACM Computing Surveys (CSUR)**, Volume 35 Issue 1

Full text available: pdf(920.16 KB) Additional Information: full citation, abstract, references, index terms

Hardware multithreading is becoming a generally applied technique in the next generation of microprocessors multithreaded processors are announced by industry or already into production in the areas of high-performa microprocessors, media, and network processors.A multithreaded processor is able to pursue two or more thr parallel within the processor pipeline. The contexts of two or more threads of control are often stored in separ register sets. Unused i ...

**Keywords**: Blocked multithreading, interleaved multithreading, simultaneous multithreading

6 Performance comparison of ILP machines with cycle time evaluation
Tetsuya Hara, Hideki Ando, Chikako Nakanishi, Masao Nakaya
May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual internation on Computer architecture**, Volume 24 Issue 2

Full text available: pdf(1.48 MB) Additional Information: full citation, abstract, references, citings, index terms

Many studies have investigated performance improvement through exploiting instruction-level parallelism (ILP particular architecture. Unfortunately, these studies indicate performance improvement using the number of c required to execute a program, but do not quantitatively estimate the penalty imposed on the cycle time from Since the performance of a microprocessor must be measured by its execution time, a cycle time evaluation i as a cy ...

7 Enhancing software reliability with speculative threads
Jeffrey Oplinger, Monica S. Lam
October 2002 **Proceedings of the 10th international conference on Architectural support for programm and operating systems**, Volume 37 , 36 , 30 Issue 10 , 5 , 5

Full text available: pdf(1.47 MB) Additional Information: full citation, abstract, references, citings

This paper advocates the use of a monitor-and-recover programming paradigm to enhance the reliability of so proposes an architectural design that allows software and hardware to cooperate in making this paradigm mo easier to program.We propose that programmers write monitoring functions assuming simple sequential exec Our architecture speeds up the computation by executing the monitoring functions speculatively in parallel wi computation. For ...

8 Value-based clock gating and operation packing: dynamic strategies for improving processor power an
David Brooks, Margaret Martonosi
May 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 2

Full text available: pdf(210.51 KB) Additional Information: full citation, abstract, references, citings, index terms

The large address space needs of many current applications have pushed processor designs toward 64-bit wo Although full 64-bit addresses and operations are indeed sometimes needed, arithmetic operations on much s are still more common. In fact, another instruction set trend has been the introduction of instructions geared operations on 16-bit quantities. For examples, most major processors now include instruction set support for operation ...

9 System-level power optimization: techniques and tools
Luca Benini, Giovanni de Micheli
April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Full text available: pdf(385.22 KB) Additional Information: full citation, abstract, references, citings, index terms

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic sytems hardware platform and software layers. We consider the three major constituents of hardware that consume e computation, communication, and storage units, and we review methods of reducing their energy consumptio models for analyzing the energy cost of software, and methods for energy-efficient software design and comp survery ...

**10** Resource allocation in a high clock rate microprocessor

Michael Upton, Thomas Huff, Trevor Mudge, Richard Brown

November 1994 **Proceedings of the sixth international conference on Architectural support for program languages and operating systems**, Volume 29 , 28 Issue 11 , 5

Full text available: pdf(1.10 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses the design of a high clock rate (300MHz) processor. The architecture is described, and th design are explained. The performance of three processor models is evaluated using trace-driven simulation. used to estimate the resources required to build processors with varying sizes of on-chip memories, in both s issue models. Recommendations are then made to increase the effectiveness of each of the models.

**Keywords**: decoupled architecture, floating point latencies, nonblocking cache, pipelining, prefetching, resou superscalar

**11** Integrated predicated and speculative execution in the IMPACT EPIC architecture

David I. August, Daniel A. Connors, Scott A. Mahlke, John W. Sias, Kevin M. Crozier, Ben-Chung Cheng, Patrick B. Olaniran, Wen-mei W. Hwu

April 1998 **ACM SIGARCH Computer Architecture News , Proceedings of the 25th annual internation on Computer architecture**, Volume 26 Issue 3

Full text available: pdf(1.60 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

Explicitly Parallel Instruction Computing (EPIC) architectures require the compiler to express program instruct parallelism directly to the hardware. EPIC techniques which enable the compiler to represent control speculati dependence speculation, and predication have individually been shown to be very effective. However, these te not been studied in combination with each other. This paper presents the IMPACT EPIC Architecture to addres involved in design ...

**12** WISQ: a restartable architecture using queues

A. R. Pleszkun, J. R. Goodman, W. C. Hsu, R. T. Joersz, G. Bier, P. Woest, P. B. Schechter

June 1987 **Proceedings of the 14th annual international symposium on Computer architecture**

Full text available: pdf(1.14 MB) Additional Information: full citation, abstract, references, citings, index terms

In this paper, the WISQ architecture is described. This architecture is designed to achieve high performance b compiler technology and using a highly segmented pipeline. By having a highly segmented pipeline, a very-hi can be used. Since a highly segmented pipeline will require relatively long pipelines, a way must be provided effects of pipeline bubbles that are formed due to data and control dependencies. It is also important to provi

**13** Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborativ**

Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time dia used to obtain a better understanding of the execution of the application. The visualization tool we use is Poe developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the desired overview of the application. In our experience, such tools display repeated occurrences of non-triv

**14** Computing curricula 2001

September 2001 **Journal on Educational Resources in Computing (JERIC)**

Full text available: pdf(613.63 KB) html(2.78 Additional Information: full citation, references, citings, index terms

KB)

15 Specification and verification of pipelining in the ARM2 RISC microprocessor
James K. Huggins, David Van Campenhout
October 1998 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 3 Issue 4
Full text available: pdf(129.88 KB) Additional Information: full citation, abstract, references, citings, index terms

Gurevich Abstract State Machines (ASMs) provide a sound mathematical basis for the specification and verific An application of the ASM methodology to the verification of a pipelined microprocessor (an ARM2 implementa described. Both the sequential execution model and final pipelined model are formalized using ASMs. A series models are introduced that gradually expose the complications of pipelining. The first intermediate model is p t ...

**Keywords**: ARM processor, abstract state machines, design verification, formal verification, pipelined process

16 Unconstrained speculative execution with predicated state buffering
Hideki Ando, Chikako Nakanishi, Tetsuya Hara, Masao Nakaya
May 1995 **ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual internatio on Computer architecture**, Volume 23 Issue 2
Full text available: pdf(1.50 MB) Additional Information: full citation, abstract, references, citings, index terms

Speculative execution is execution of instructions before it is known whether these instructions should be exe based speculative execution has the potential to achieve both a high instruction per cycle rate and high clock compiler-based approaches, however, have greatly limited instruction scheduling due to a limited ability to ha of speculative execution. Significant performance improvement is, thus, difficult in non-numerical applications

17 Semi-formal test generation with genevieve
Julia Dushina, Mike Benjamin, Daniel Geist
June 2001 **Proceedings of the 38th conference on Design automation**
Full text available: pdf(146.93 KB) Additional Information: full citation, abstract, references, index terms

This paper describes the first application of the Genevieve test generation methodology. The Genevieve appro for-mal techniques derived from "model-checking" to generate test suites for specific behaviours of the design "interest-ing" behaviour is claimed to be unreachable. If a path from an ini-tial state to the state of interest d counter-example is generated. The sequence of states specifies a test for the desired behaviour. ...

18 Efficient superscalar performance through boosting
Michael D. Smith, Mark Horowitz, Monica S. Lam
September 1992 **ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectu programming languages and operating systems**, Volume 27 Issue 9
Full text available: pdf(1.63 MB) Additional Information: full citation, abstract, references, citings, index terms

The foremost goal of superscalar processor design is to increase performance through the exploitation of instr parallelism (ILP). Previous studies have shown that speculative execution is required for high instruction per c in non-numerical applications. The general trend has been toward supporting speculative execution in complic dynamically-scheduled processors. Performance, though, is more than just a high IPC rate; it also depends up count ...

19 Speculative multithreaded processors
Pedro Marcuello, Antonio González, Jordi Tubella
July 1998 **Proceedings of the 12th international conference on Supercomputing**
Full text available: pdf(1.24 MB) Additional Information: full citation, references, citings, index terms

**Keywords**: control speculation, data dependence speculation, data speculation, dynamically scheduled proce multithreaded processors

**20** Architecture 2: Dual path instruction processing

Juan L. Aragón, José González, Antonio González, James E. Smith

June 2002 **Proceedings of the 16th international conference on Supercomputing**

Full text available: pdf(332.19 KB) Additional Information: full citation, abstract, references, index terms

The reasons for performance losses due to conditional branch mispredictions are first studied. Branch mispred are broken into three categories: pipeline-fill penalty, window-fill penalty, and serialization penalty. The first a produce most of the performance loss, but the second is also significant. Previously proposed dual (or multi) p methods attempt to reduce all three penalties, but these methods are also quite complex. Most of the comple

**Keywords:** branch misprediction penalty, confidence estimation, dual path processing, pre-scheduling

| | Type | L # | Hits | Search Text | DBs | Time Stamp | Comments |
|---|---|---|---|---|---|---|---|
| 1 | BRS | L1 | 15 | (branch$5 near3 instruction WITH processing near3 unit) SAME clock adj cycle | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:12 | |
| 2 | BRS | L2 | 1 | 1 and address$3 with point$3 with register | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:13 | |
| 3 | BRS | L3 | 4 | 1 and address$3 with point$3 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:08 | |
| 4 | BRS | L4 | 13 | 1 and point$3 with register | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:09 | |
| 5 | BRS | L5 | 6 | 4 and valid$5 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:14 | |
| 6 | BRS | L6 | 395231 | "6" and check | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:09 | |
| 7 | BRS | L7 | 2 | 5 and check | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 17:01 | |

| | Type | L # | Hits | Search Text | DBs | Time Stamp | Comments |
|---|---|---|---|---|---|---|---|
| 8 | BRS | L8 | 50 | (branch$5 near3 instruction and processing near3 unit) SAME clock adj cycle | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:13 | |
| 9 | BRS | L9 | 35 | 8 not 1 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:13 | |
| 10 | BRS | L10 | 16 | 9 and address$3 with point$3 with register | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:14 | |
| 11 | BRS | L11 | 15 | 10 and valid$5 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:14 | |
| 12 | BRS | L12 | 9 | 11 and check | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 16:15 | |
| 13 | BRS | L13 | 2 | guard$3 SAME check SAME valid$5 SAME branch$5 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 17:03 | |
| 14 | BRS | L14 | 179 | guard$3 and check and valid$5 SAME branch$5 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 17:03 | |

| | Type | L # | Hits | Search Text | DBs | Time Stamp | Comments |
|---|---|---|---|---|---|---|---|
| 15 | BRS | L15 | 134 | 14 and register | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 17:03 | |
| 16 | BRS | L16 | 4 | 15 and adress$5 | USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB | 2004/09/10 17:03 | |